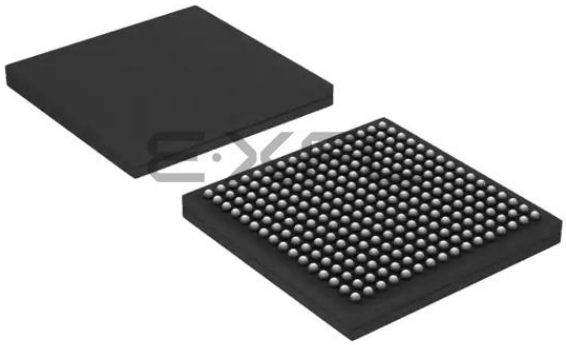

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	Coldfire V2
Core Size	32-Bit Single-Core
Speed	66MHz
Connectivity	CANbus, EBI/EMI, I²C, SPI, UART/USART
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	142
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	64K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	256-LBGA
Supplier Device Package	256-MAPBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcf5216cvm66j

- Detection of breaks originating in the middle of a character
- Start/end break interrupt/status

1.1.11 DMA Timers (DTIM0-DTIM3)

There are four independent, DMA-transfer-generating 32-bit timers (DTIM0, DTIM1, DTIM2, DTIM3) on the MCF5282. Each timer module incorporates a 32-bit timer with a separate register set for configuration and control. The timers can be configured to operate from the system clock or from an external clock source using one of the DTIN*x* signals. If the system clock is selected, it can be divided by 16 or 1. The selected clock is further divided by a user-programmable 8-bit prescaler which clocks the actual timer counter register (TCR*n*). Each of these timers can be configured for input capture or reference compare mode. By configuring the internal registers, each timer may be configured to assert an external signal, generate an interrupt on a particular event, or cause a DMA transfer.

1.1.12 General-Purpose Timers (GPTA/GPTB)

The two general-purpose timers (GPTA and GPTB) are 4-channel timer modules. Each timer consists of a 16-bit programmable counter driven by a 7-stage programmable prescaler. Each of the four channels for each timer can be configured for input capture or output compare. Additionally, one of the channels, channel 3, can be configured as a pulse accumulator.

A timer overflow function allows software to extend the timing capability of the system beyond the 16-bit range of the counter. The input capture and output compare functions allow simultaneous input waveform measurements and output waveform generation. The input capture function can capture the time of a selected transition edge. The output compare function can generate output waveforms and timer software delays. The 16-bit pulse accumulator can operate as a simple event counter or a gated time accumulator.

1.1.13 Periodic Interrupt Timers (PIT0-PIT3)

The four periodic interrupt timers (PIT0, PIT1, PIT2, PIT3) are 16-bit timers that provide precise interrupts at regular intervals with minimal processor intervention. Each timer can either count down from the value written in its PIT modulus register, or it can be a free-running down-counter.

1.1.14 Software Watchdog Timer

The watchdog timer is a 16-bit timer that facilitates recovery from runaway code. The watchdog counter is a free-running down-counter that generates a reset on underflow. To prevent a reset, software must periodically restart the countdown.

1.1.15 Phase Locked Loop (PLL)

The clock module contains a crystal oscillator (OSC), phase-locked loop (PLL), reduced frequency divider (RFD), status/control registers, and control logic. To improve noise immunity, the PLL and OSC have their own power supply inputs, VDDPLL and VSSPLL. All other circuits are powered by the normal supply pins, VDD and VSS.

1.1.16 DMA Controller

The Direct Memory Access (DMA) controller module provides an efficient way to move blocks of data with minimal processor interaction. The DMA module provides four channels (DMA0–DMA3) that allow

As with change or use stalls between accumulators and general-purpose registers, introducing intervening instructions that do not reference the busy register can reduce or eliminate sequence-related store-MAC instruction stalls. A major benefit of the EMAC is the addition of three accumulators to minimize stalls caused by exchanges between accumulator(s) and general-purpose registers.

3.3.4 Data Representation

MACSR[S/U,F/I] selects one of the following three modes, where each mode defines a unique operand type:

1. Two's complement signed integer: In this format, an N-bit operand value lies in the range $-2^{(N-1)} \leq$ operand $\leq 2^{(N-1)} - 1$. The binary point is right of the lsb.
2. Unsigned integer: In this format, an N-bit operand value lies in the range $0 \leq$ operand $\leq 2^{N} - 1$. The binary point is right of the lsb.
3. Two's complement, signed fractional: In an N-bit number, the first bit is the sign bit. The remaining bits signify the first N-1 bits after the binary point. Given an N-bit number, $a_{N-1}a_{N-2}a_{N-3}... a_2a_1a_0$, its value is given by the equation in Equation 3-3.

$$\text{value} = -(1 \cdot a_{N-1}) + \sum_{i=0}^{N-2} 2^{-(i+1-N)} \cdot ai \qquad \textit{Eqn. 3-3}$$

This format can represent numbers in the range $-1 \leq$ operand $\leq 1 - 2^{(N-1)}$.

For words and longwords, the largest negative number that can be represented is -1, whose internal representation is 0x8000 and 0x8000_0000, respectively. The largest positive word is 0x7FFF or $(1 - 2^{-15})$; the most positive longword is 0x7FFF_FFFF or $(1 - 2^{-31})$.

3.3.5 MAC Opcodes

MAC opcodes are described in the *ColdFire Programmer's Reference Manual*.

Remember the following:

- Unless otherwise noted, the value of MACSR[N,Z] is based on the result of the final operation that involves the product and the accumulator.
- The overflow (V) flag is managed differently. It is set if the complete product cannot be represented as a 40-bit value (this applies to 32 × 32 integer operations only) or if the combination of the product with an accumulator cannot be represented in the given number of bits. The EMAC design includes an additional product/accumulation overflow bit for each accumulator that are treated as sticky indicators and are used to calculate the V bit on each MAC or MSAC instruction. See Section 3.2.1, "MAC Status Register (MACSR)".
- For the MAC design, the assembler syntax of the MAC (multiply and add to accumulator) and MSAC (multiply and subtract from accumulator) instructions does not include a reference to the single accumulator. For the EMAC, assemblers support this syntax and no explicit reference to an accumulator is interpreted as a reference to ACC0. Assemblers also support syntaxes where the destination accumulator is explicitly defined.

4.3.5 Cache Miss Fetch Algorithm/Line Fills

As discussed in Section 4.1.2, “Introduction,” the cache hardware includes a 16-byte, line-fill buffer for providing temporary storage for the last fetched line.

With the cache enabled as defined by CACR[CENB], a cacheable fetch that misses in the tag memory and the line-fill buffer generates an external fetch. For data misses, the size of the external fetch is always 16 bytes. For instruction misses, the size of the external fetch is determined by the value contained in the 2-bit CLNF field of the CACR and the miss address. Table 4-6 shows the relationship between the CLNF bits, the miss address, and the size of the external fetch.

Table 4-6. Initial Fetch Offset vs. CLNF Bits

CLNF[1:0]	Longword Address Bits[3:2]			
	00	01	10	11
00	Line	Line	Line	Longword
01	Line	Line	Longword	Longword
1X	Line	Line	Line	Line

Depending on the runtime characteristics of the application and the memory response speed, overall performance may be increased by programming the CLNF bits to values 00 or 01.

For all cases of a line-sized fetch, the critical longword defined by bits [3:2] of the miss address is accessed first followed by the remaining three longwords that are accessed by incrementing the longword address in a modulo-16 fashion as shown below:

```
if miss address[3:2] = 00
   fetch sequence = 0x0, 0x4, 0x8, 0xC
if miss address[3:2] = 01
   fetch sequence = 0x4, 0x8, 0xC, 0x0
if miss address[3:2] = 10
   fetch sequence = 0x8, 0xC, 0x0, 0x4
if miss address[3:2] = 11
   fetch sequence = 0xC, 0x0, 0x4, 0x8
```

After an external fetch has been initiated and the data is loaded into the line-fill buffer, the cache maintains a special most-recently-used indicator that tracks the contents of the associated line-fill buffer versus its corresponding cache location. At the time of the miss, the hardware indicator is set, marking the line-fill buffer as most recently used. If a subsequent access occurs to the cache location defined by bits [10:4] (or bits [9:4] for split configurations of the fill buffer address), the data in the cache memory array is now most recently used, so the hardware indicator is cleared. In all cases, the indicator defines whether the contents of the line-fill buffer or the memory data array are most recently used. At the time of the next cache miss, the contents of the line-fill buffer are written into the memory array if the entire line is present, and the line-fill buffer data is most recently used compared to the memory array.

Generally, longword references are used for sequential instruction fetches. If the processor branches to an odd word address, a word-sized instruction fetch is generated.

6.3.1 CFM Configuration Field

The CFM configuration field comprises 24 bytes of reserved array memory space that determines the module protection and access restrictions out of reset. Data to secure the Flash from unauthorized access is also stored in the CFM configuration field. Table 6-1 describes each byte used in this field.

Table 6-1. CFM Configuration Field

Address Offset (from array base address)	Size in Bytes	Description
0x0000_0400–0x0000_0407	8	Back door comparison key
0x0000_0408–0x0000_040B	4	Flash program/erase sector protection Blocks 0H/0L (see Section 6.3.4.4, “CFM Protection Register (CFMPROT)”)
0x0000_040C–0x0000_040F	4	Flash supervisor/user space restrictions Blocks 0H/0L (see Section 6.3.4.5, “CFM Supervisor Access Register (CFMSACC)”)
0x0000_0410–0x0000_0413	4	Flash program/data space restrictions Blocks 0H/0L (see Section 6.3.4.6, “CFM Data Access Register (CFMDACC)”)
0x0000_0414–0x0000_0417	4	Flash security longword (see Section 6.3.4.3, “CFM Security Register (CFMSEC)”)

6.3.2 Flash Base Address Register (FLASHBAR)

The configuration information in the Flash base address register (FLASHBAR) controls the operation of the Flash module.

- The FLASHBAR holds the base address of the Flash. The MOVEC instruction provides write-only access to this register.
- The FLASHBAR can be read or written from the debug module in a similar manner.
- All undefined bits in the register are reserved. These bits are ignored during writes to the FLASHBAR, and return zeroes when read from the debug module.
- The back door enable bit, FLASHBAR[BDE], is cleared at reset, disabling back door access to the Flash.
- The FLASHBAR valid bit is programmed according to the chip mode selected at reset (see Chapter 27, “Chip Configuration Module (CCM)” for more details). All other bits are unaffected.

The FLASHBAR register contains several control fields. These fields are shown in Figure 6-3

NOTE

The default value of the FLASHBAR is determined by the chip configuration selected at reset (see Chapter 27, “Chip Configuration Module (CCM)” for more information). If external boot mode is used, then the FLASHBAR located in the processor’s CPU space will be invalid and it must be initialized with the valid bit set before the CPU (or modules) can access the on-chip Flash.

6.3.4.4 CFM Protection Register (CFMPROT)

The CFMPROT specifies which Flash logical sectors are protected from program and erase operations.

	31															16
Field	PROT															
Reset	See Note															
R/W	R/W															

	15															0
Field	PROT															
Reset	See Note															
R/W	R/W															
Address	IPSBAR + 0x1D_0010															

Note: The CFMPROT register is loaded at reset from the Flash Program/Erase Sector Protection longword stored at the array base address + 0x0000_0400.

Figure 6-7. CFM Protection Register (CFMPROT)

The CFMPROT register is always readable and only writeable when LOCK = 0. To change which logical sectors are protected on a temporary basis, write CFMPROT with a new value after the LOCK bit in CFMCR has been cleared. To change the value of CFMPROT that will be loaded on reset, the protection byte in the Flash configuration field must first be temporarily unprotected using the method just described before reprogramming the protection bytes. Then the Flash Protection longword at offset 0x1D_0400 must be written with the desired value.

Table 6-7. CFMPROT Field Descriptions

Bits	Name	Description
31–0	PROT[31:0]	Sector protection. Each Flash logical sector can be protected from program and erase operations by setting its corresponding PROT bit. 1 Logical sector is protected. 0 Logical sector is not protected.

The CFMPROT controls the protection of thirty-two 16-Kbyte Flash logical sectors in the 512-Kbyte Flash array. Figure 6-8 shows the association between each bit in the CFMPROT and its corresponding logical sector.

NOTE

Since the MCF5281 and MCF5214 devices contain a 256-Kbyte Flash, only CFMPROT[15:0] is used.

6.5.1 Back Door Access

If the KEYEN bit is set, security can be bypassed by:

1. Setting the KEYACC bit in the CFM configuration register (CFMMCR).
2. Writing the correct 8-byte back door comparison key to the CFM array at addresses 0x0000_0400 to 0x0000_0407. This operation must consist of two 32-bit writes to address 0x0000_0400 and 0x0000_0404 in that order. The two back door write cycles can be separated by any number of bus cycles.
3. Clearing the KEYACC bit.
4. If all 8 bytes written match the array contents at addresses 0x0000_0400 to 0x0000_0407, then security is bypassed until the next reset.

NOTE

The security of the Flash as defined by the Flash security longword at address 0x0000_0414 is not changed by the back door method of unsecuring the device. After the next reset the device is again secured and the same back door key remains in effect unless changed by program or erase operations. The back door method of unsecuring the device has no effect on the program and erase protections defined by the CFM protection register (CFMPROT).

6.5.2 Erase Verify Check

Security can be disabled by verifying that the CFM array is blank. If required, the mass erase command can be executed for each pair of Flash physical blocks that comprise the array. The erase verify command must then be executed for all Flash physical blocks within the array. The CFM will be unsecured if the erase verify command determines that the entire array is blank. After the next reset, the security state of the CFM will be determined by the Flash security longword, which, after being erased, will read 0xffff_ffff, thus unsecuring the module.

6.6 Reset

The CFM array is not accessible for any operations via the address and data buses during reset. If a reset occurs while any command is in progress that command will immediately abort. The state of any longword being programmed or any erase pages/physical blocks being erased is not guaranteed.

6.7 Interrupts

The CFM module can request an interrupt when all commands are completed or when the address, data, and command buffers are empty. Table 6-14 shows the CFM interrupt mechanism.

Table 6-14. CFM Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable
Command, data and address buffers empty	CBEIF (CFMUSTAT)	CBEIE (CFMCR)

Table 8-12. GPACR Field Descriptions

Bits	Name	Description
7	LOCK	This bit, once set, prevents subsequent writes to the GPACR. Any attempted write to the GPACR generates an error termination and the contents of the register are not affected. Only a system reset clears this flag.
6–4	—	Reserved, should be cleared.
3–0	ACCESS_CTRL	This 4-bit field defines the access control for the given memory region. The encodings for this field are shown in Table 8-13.

At reset, these on-chip modules are configured to have only supervisor read/write access capabilities. Bit encodings for the ACCESS_CTRL field in the GPACR are shown in Table 8-13. Table 8-14 shows the memory space protected by the GPACRs and the modules mapped to these spaces.

Table 8-13. GPACR ACCESS_CTRL Bit Encodings

Bits	Supervisor Mode	User Mode
0000	Read / Write	No Access
0001	Read	No Access
0010	Read	Read
0011	Read	No Access
0100	Read / Write	Read / Write
0101	Read / Write	Read
0110	Read / Write	Read / Write
0111	No Access	No Access
1000	Read / Write / Execute	No Access
1001	Read / Execute	No Access
1010	Read / Execute	Read / Execute
1011	Execute	No Access
1100	Read / Write / Execute	Read / Write / Execute
1101	Read / Write / Execute	Read / Execute
1110	Read / Write	Read
1111	Read / Write / Execute	Execute

Table 14-4. Pin Reset States at Reset (Single-Chip Mode) (continued)

Signal	Reset	I/O
Debug Support Signals		
JTAG_EN	—	I
DSCLK/$\overline{\text{TRST}}$	—	I
$\overline{\text{BKPT}}$/TMS	—	I
DSI/TDI	—	I
DSO/TDO	High	O
TCLK	—	I
DDATA[3:0]	DDATA{3:0]	O
PST[3:0]	PST[3:0]	O

14.1.2 External Boot Mode

When booting from external memory, the address bus, data bus, and bus control signals will default to their bus functionalities as shown in Table 14-5. As in single-chip mode, the signals listed in Table 14-4 will operate as described above. All other signals will default to GPIO inputs.

Table 14-5. Default Signal Functions After System Reset (External Boot Mode)

Signal	Reset	I/O
A[23:0]	A[23:0]	O
D[31:0]	—	I/O
$\overline{\text{BS}}$[3:0]	High	O
$\overline{\text{OE}}$	High	O
$\overline{\text{TA}}$	—	I
$\overline{\text{TEA}}$	—	I
R/$\overline{\text{W}}$	High	O
SIZ[1:0]	High	O
$\overline{\text{TS}}$	High	O
$\overline{\text{TIP}}$	High	O
$\overline{\text{CS}}$[6:0]	High	O

14.2 External Signals

The following sections describe the external signals on the device.

14.2.1 External Interface Module (EIM) Signals

These signals are used for doing transactions on the external bus.

14.2.10.4 Request-to-Send ($\overline{\text{URTS}}$[1:0])

The $\overline{\text{URTS}}$[1:0] signals are automatic request to send outputs from the UART modules. $\overline{\text{URTS}}$[1:0] can also be configured to be asserted and negated as a function of the Rx FIFO level.

The $\overline{\text{URTS}}$[1:0] outputs are each offered as secondary functions on four pins: DTIN3, DTOUT3, DTIN1 and DTOUT1.

14.2.11 General Purpose Timer Signals

These pins provide the external interface to the general purpose timer functions.

14.2.11.1 GPTA[3:0]

These pins provide the external interface to the timer A functions.

These pins can also be configured as GPIO PTA[3:0].

14.2.11.2 GPTB[3:0]

These pins provide the external interface to the timer B functions.

These pins can also be configured as GPIO PTB[3:0].

14.2.11.3 External Clock Input (SYNCA/SYNCB)

These pins are used to clear the clock for each of the two timers, and are provided as a means of synchronization to externally clocked or timed events.

14.2.12 DMA Timer Signals

This section describes the signals of the four DMA timer modules.

14.2.12.1 DMA Timer 0 Input (DTIN0)

The DMA timer 0 input (DTIN0) can be programmed to cause events to occur in DMA timer 0. It can either clock the event counter or provide a trigger to the timer value capture logic.

This pin can also be configured as GPIO PTD1, secondary function $\overline{\text{UCTS1}}$, or secondary function $\overline{\text{UCTS0}}$.

14.2.12.2 DMA Timer 0 Output (DTOUT0)

The programmable DMA timer output (DTOUT0) pulse or toggle on various timer events.

This pin can also be configured as GPIO PTD0, secondary function $\overline{\text{UCTS1}}$, or secondary function $\overline{\text{UCTS0}}$.

14.2.12.3 DMA Timer 1 Input (DTIN1)

The DMA timer 1 input (DTIN1) can be programmed to cause events to occur in DMA timer 1. This can either clock the event counter or provide a trigger to the timer value capture logic.

If DSIZE is another size, data writes are optimized to write the largest size allowed based on the address, but not exceeding the configured size.

16.5.4.2 Bandwidth Control

Bandwidth control makes it possible to force the DMA off the bus to allow access to another device. DCR*n*[BWC] provides seven levels of block transfer sizes. If the BCR*n* decrements to a multiple of the decode of the BWC, the DMA bus request negates until the bus cycle terminates. If a request is pending, the arbiter may then pass bus mastership to another device. If auto-alignment is enabled, DCR*n*[AA] = 1, the BCR*n* may skip over the programmed boundary, in which case, the DMA bus request is not negated.

If BWC = 000, the request signal remains asserted until BCR*n* reaches zero. DMA has priority over the core. Note that in this scheme, the arbiter can always force the DMA to relinquish the bus. See Section 8.5.3, “Bus Master Park Register (MPARK).”

16.5.5 Termination

An unsuccessful transfer can terminate for one of the following reasons:

- Error conditions—When the processor encounters a read or write cycle that terminates with an error condition, DSR*n*[BES] is set for a read and DSR*n*[BED] is set for a write before the transfer is halted. If the error occurred in a write cycle, data in the internal holding register is lost.
- Interrupts—If DCR*n*[INT] is set, the DMA drives the appropriate internal interrupt signal. The processor can read DSR*n* to determine whether the transfer terminated successfully or with an error. DSR*n*[DONE] is then written with a one to clear the interrupt and the DONE and error bits.

IPSBAR 0x15_0004 (PCNTR0)
Offset: 0x16_0004 (PCNTR1)
0x17_0004 (PCNTR2)
0x18_0004 (PCNTR3)

Access: User read only

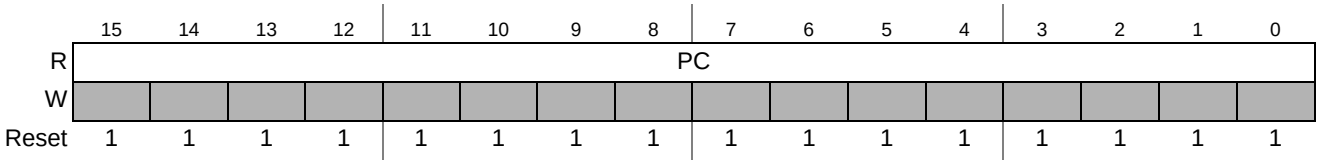

Figure 19-4. PIT Count Register (PCNTR*n*)

Table 19-5. PCNTR*n* Field Descriptions

Field	Description
15–0 PC	Counter value. Reading this field with two 8-bit reads is not guaranteed coherent. Writing to PCNTR*n* has no effect, and write cycles are terminated normally.

19.3 Functional Description

This section describes the PIT functional operation.

19.3.1 Set-and-Forget Timer Operation

This mode of operation is selected when the RLD bit in the PCSR register is set.

When PIT counter reaches a count of 0x0000, PIF flag is set in PCSR*n*. The value in the modulus register loads into the counter, and the counter begins decrementing toward 0x0000. If the PCSR*n*[PIE] bit is set, the PIF flag issues an interrupt request to the CPU.

When the PCSR*n*[OVW] bit is set, the counter can be directly initialized by writing to PMR*n* without having to wait for the count to reach 0x0000.

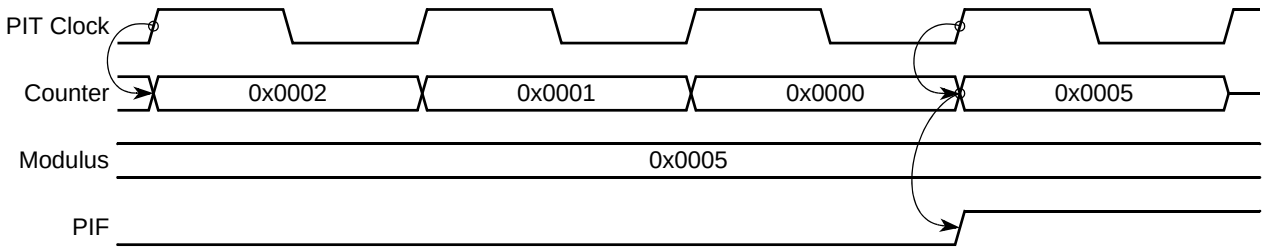

Figure 19-5. Counter Reloading from the Modulus Latch

19.3.2 Free-Running Timer Operation

This mode of operation is selected when the PCSR*n*[RLD] bit is clear. In this mode, the counter rolls over from 0x0000 to 0xFFFF without reloading from the modulus latch and continues to decrement.

When the counter reaches a count of 0x0000, PCSR*n*[PIF] flag is set. If the PCSR*n*[PIE] bit is set, PIF flag issues an interrupt request to the CPU.

20.5.19 GPT Port Data Direction Register (GPTDDR)

	7	6	5	4	3	2	1	0
Field	—				DDRT			
GPT Function	—				IC/OC			
Pulse Accumulator Function	—				PAI	—		
Reset	0000_0000							
R/W	R/W							
Address	IPSBAR + 0x1A_001E, 0x1B_001E							

Figure 20-21. GPT Port Data Direction Register (GPTDDR)

Table 20-22. GPTDDR Field Descriptions

Bit(s)	Name	Description
7–4	—	Reserved, should be cleared.
3–0	DDRT	Control the port logic of PORTT*n*. Reset clears the PORTT*n* data direction register, configuring all GPT port pins as inputs. These bits are read anytime, write anytime. 1 Corresponding pin configured as output 0 Corresponding pin configured as input

20.6 Functional Description

The General Purpose Timer (GPT) module is a 16-bit, 4-channel timer with input capture and output compare functions and a pulse accumulator.

20.6.1 Prescaler

The prescaler divides the module clock by 1, 2, 4, 8, 16, 32, 64, or 128. The GPTSCR2[PR] bits select the prescaler divisor.

20.6.2 Input Capture

Clearing an I/O select bit, IOS*n*, configures channel *n* as an input capture channel. The input capture function captures the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the timer transfers the value in the GPT counter into the GPT channel registers, GPTC*n*.

The minimum pulse width for the input capture input is greater than two module clocks.

The input capture function does not force data direction. The GPT port data direction register controls the data direction of an input capture pin. Pin conditions such as rising or falling edges can trigger an input capture only on a pin configured as an input.

An input capture on channel *n* sets the C*n*F flag. The C*n*I bit enables the C*n*F flag to generate interrupt requests.

NOTE

When the GPT channel 3 registers contain 0xFFFF and TCRE is set, TOF does not get set even though the GPT counter registers go from 0xFFFF to 0x0000.

When the fast flag clear all bit, GPTSCR1[TFFCA], is set, any access to the GPT counter registers clears GPT flag register 2.

When TOF is set, it does not inhibit future overflow events.

If the free run/restart bit (DTMR*n*[FRR]) is set, a new count starts. If it is clear, the timer keeps running.

21.3.4 Output Mode

When a timer reaches the reference value selected by DTRR, it can send an output signal on DTOUT*n*. DTOUT*n* can be an active-low pulse or a toggle of the current output, as selected by the DTMR*n*[OM] bit.

21.4 Initialization/Application Information

The general-purpose timer modules typically, but not necessarily, follow this program order:

- The DTMR*n* and DTXMR*n* registers are configured for the desired function and behavior.
  - Count and compare to a reference value stored in the DTRR*n* register
  - Capture the timer value on an edge detected on DTIN*n*
  - Configure DTOUT*n* output mode
  - Increment counter by 1 or by 65,537 (16-bit mode)
  - Enable/disable interrupt or DMA request on counter reference match or capture edge
- The DTMR*n*[CLK] register is configured to select the clock source to be routed to the prescaler.
  - Internal bus clock (can be divided by 1 or 16)
  - DTIN*n*, the maximum value of DTIN*n* is 1/5 of the internal bus clock, as described in the device's electrical characteristics

NOTE

DTIN*n* may not be configured as a clock source when the timer capture mode is selected or indeterminate operation results.

- The 8-bit DTMR*n*[PS] prescaler value is set.
- Using DTMR*n*[RST], counter is cleared and started.
- Timer events are managed with an interrupt service routine, a DMA request, or by a software polling mechanism.

21.4.1 Code Example

The following code provides an example of how to initialize and use DMA Timer0 for counting time-out periods.

```
DTMR0 EQU IPSBARx+0x400 ;Timer0 mode register
DTMR1 EQU IPSBARx+0x440 ;Timer1 mode register
DTRR0 EQU IPSBARx+0x404 ;Timer0 reference register
DTRR1 EQU IPSBARx+0x444 ;Timer1 reference register
DTCR0 EQU IPSBARx+0x408 ;Timer0 capture register
DTCR1 EQU IPSBARx+0x448 ;Timer1 capture register
DTCN0 EQU IPSBARx+0x40C ;Timer0 counter register
DTCN1 EQU IPSBARx+0x44C ;Timer1 counter register
DTER0 EQU IPSBARx+0x403 ;Timer0 event register
DTER1 EQU IPSBARx+0x443 ;Timer1 event register
* TMR0 is defined as: *
*[PS] = 0xFF,     divide clock by 256
```

Chapter 22
Queued Serial Peripheral Interface (QSPI)

22.1 Introduction

This chapter describes the queued serial peripheral interface (QSPI) module.

22.1.1 Block Diagram

Figure 22-1 illustrates the QSPI module.

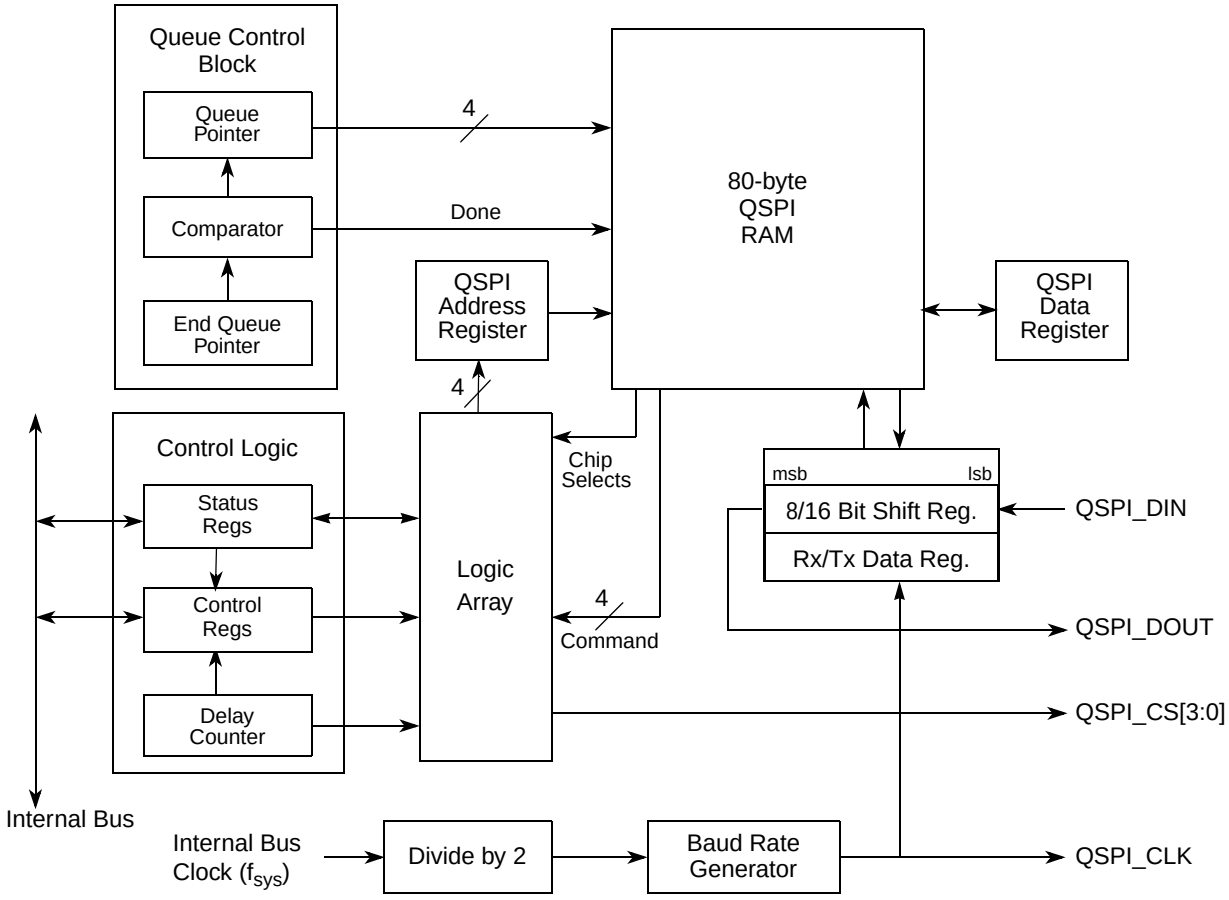

Figure 22-1. QSPI Block Diagram

Table 23-5. USR*n* Field Descriptions (continued)

Field	Description
1 FFULL	FIFO full. 0 The FIFO is not full but may hold up to two unread characters. 1 A character was received and the receiver FIFO is now full. Any characters received when the FIFO is full are lost.
0 RXRDY	Receiver ready. 0 The CPU has read the receive buffer and no characters remain in the FIFO after this read. 1 One or more characters were received and are waiting in the receive buffer FIFO.

23.3.4 UART Clock Select Registers (UCSR*n*)

The UCSRs select an external clock on the DTIN input (divided by 1 or 16) or a prescaled internal bus clock as the clocking source for the transmitter and receiver. See Section 23.4.1, “Transmitter/Receiver Clock Source.” The transmitter and receiver can use different clock sources. To use the internal bus clock for both, set UCSR*n* to 0xDD.

IPSBAR 0x00_0204 (UCSR0) Access: User write-only
Offset: 0x00_0244 (UCSR1)
0x00_0284 (UCSR2)

	7	6	5	4	3	2	1	0
R								
W	RCS				TCS			
Reset:	See Note				See Note			

Note: The RCS and TCS reset values are set so the receiver and transmiter use the prescaled internal bus clock as their clock source.

Figure 23-6. UART Clock Select Registers (UCSR*n*)

Table 23-6. UCSR*n* Field Descriptions

Field	Description
7–4 RCS	Receiver clock select. Selects the clock source for the receiver. 1101 Prescaled internal bus clock (f_{sys}) 1110 DTIN*n* divided by 16 1111 DTIN*n*
3–0 TCS	Transmitter clock select. Selects the clock source for the transmitter. 1101 Prescaled internal bus clock (f_{sys}) 1110 DTIN*n* divided by 16 1111 DTIN*n*

23.3.5 UART Command Registers (UCR*n*)

The UCRs supply commands to the UART. Only multiple commands that do not conflict can be specified in a single write to a UCR*n*. For example, RESET TRANSMITTER and ENABLE TRANSMITTER cannot be specified in one command.

IPSBAR 0x00_020C (URB0)
Offset: 0x00_024C (URB1)
0x00_028C (URB2)

Access: User read-only

	7	6	5	4	3	2	1	0
R	RB							
W								
Reset:	1	1	1	1	1	1	1	1

Figure 23-8. UART Receive Buffer (URB*n*)

23.3.7 UART Transmit Buffers (UTB*n*)

The transmit buffers consist of the transmitter holding register and the transmitter shift register. The holding register accepts characters from the bus master if UART's USR*n*[TXRDY] is set. A write to the transmit buffer clears USR*n*[TXRDY], inhibiting any more characters until the shift register can accept more data. When the shift register is empty, it checks if the holding register has a valid character to be sent (TXRDY = 0). If there is a valid character, the shift register loads it and sets USR*n*[TXRDY] again. Writes to the transmit buffer when the UART's TXRDY is cleared and the transmitter is disabled have no effect on the transmit buffer.

Figure 23-9 shows UTB*n*. TB contains the character in the transmit buffer.

IPSBAR 0x00_020C (UTB0)
Offset: 0x00_024C (UTB1)
0x00_028C (UTB2)

Access: User write-only

	7	6	5	4	3	2	1	0
R								
W	TB							
Reset:	0	0	0	0	0	0	0	0

Figure 23-9. UART Transmit Buffer (UTB*n*)

23.3.8 UART Input Port Change Registers (UIPCR*n*)

The UIPCRs hold the current state and the change-of-state for $\overline{\text{UCTS}}n$.

IPSBAR 0x00_0210 (UIPCR0)
Offset: 0x00_0250 (UIPCR1)
0x00_0290 (UIPCR2)

Access: User read-only

	7	6	5	4	3	2	1	0
R	0	0	0	COS	1	1	1	CTS
W								
Reset:	0	0	0	0	1	1	1	$\overline{\text{UCTS}}n$

Figure 23-10. UART Input Port Changed Registers (UIPCR*n*)

3. Unmask appropriate bits in the core's status register (SR) to enable interrupts.
4. If TXRDY or RXRDY generates interrupt requests, verify that DMAREQC (in the SCM) does not also assign the UART's TXRDY and RXRDY into DMA channels.
5. Initialize interrupts in the UART, see Table 23-13.

Table 23-13. UART Interrupts

Register	Bit	Interrupt
UMR1*n*	6	RxIRQ
UIMR*n*	7	Change of State (COS)
UIMR*n*	2	Delta Break
UIMR*n*	1	RxFIFO Full
UIMR*n*	0	TXRDY

23.5.1.2 Setting up the UART to Request DMA Service

The UART is capable of generating two internal DMA request signals: transmit and receive.

The transmit DMA request signal is asserted when the TXRDY (transmitter ready) in the UART interrupt status register (UISR*n*[TXRDY]) is set. When the transmit DMA request signal is asserted, the DMA can initiate a data copy, reading the next character transmitted from memory and writing it into the UART transmit buffer (UTB*n*). This allows the DMA channel to stream data from memory to the UART for transmission without processor intervention. After the entire message has been moved into the UART, the DMA would typically generate an end-of-data-transfer interrupt request to the CPU. The resulting interrupt service routine (ISR) could query the UART programming model to determine the end-of-transmission status.

Similarly, the receive DMA request signal is asserted when the FIFO full or receive ready (FFULL/RXRDY) flag in the interrupt status register (UISR*n*[FFULL/RXRDY]) is set. When the receive DMA request signal is asserted, the DMA can initiate a data move, reading the appropriate characters from the UART receive buffer (URB*n*) and storing them in memory. This allows the DMA channel to stream data from the UART receive buffer into memory without processor intervention. After the entire message has been moved from the UART, the DMA would typically generate an end-of-data-transfer interrupt request to the CPU. The resulting interrupt service routine (ISR) should query the UART programming model to determine the end-of-transmission status. In typical applications, the receive DMA request should be configured to use RXRDY directly (and not FFULL) to remove any complications related to retrieving the final characters from the FIFO buffer.

The implementation described in this section allows independent DMA processing of transmit and receive data while continuing to support interrupt notification to the processor for $\overline{\text{CTS}}$ change-of-state and delta break error managing.

24.3.4 Acknowledge

The transmitter releases the I2C_SDA line high during the acknowledge clock pulse as shown in Figure 24-9. The receiver pulls down the I2C_SDA line during the acknowledge clock pulse so that it remains stable low during the high period of the clock pulse.

If it does not acknowledge the master, the slave receiver must leave I2C_SDA high. The master can then generate a STOP signal to abort data transfer or generate a START signal (repeated start, shown in Figure 24-10 and discussed in Section 24.3.6, “Repeated START”) to start a new calling sequence.

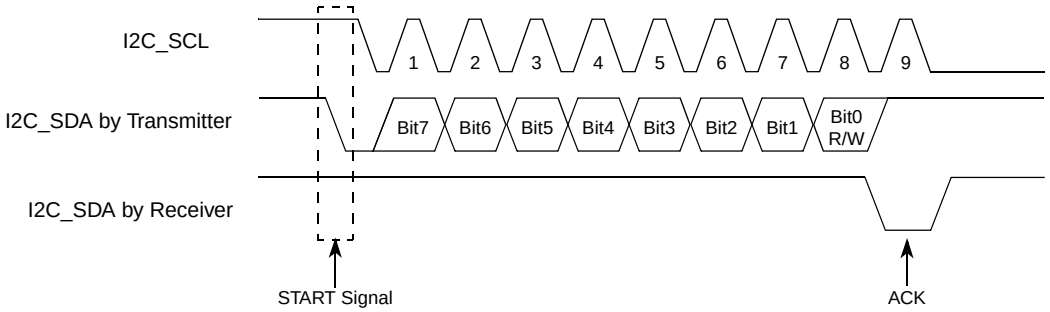

Figure 24-9. Acknowledgement by Receiver

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means end-of-data to the slave. The slave releases I2C_SDA for the master to generate a STOP or START signal (Figure 24-9).

24.3.5 STOP Signal

The master can terminate communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of I2C_SDA while I2C_SCL is at logical high (see F in Figure 24-7). The master can generate a STOP even if the slave has generated an acknowledgment, at which point the slave must release the bus. The master may also generate a START signal following a calling address, without first generating a STOP signal. Refer to Section 24.3.6, “Repeated START.”

24.3.6 Repeated START

A repeated START signal is a START signal generated without first generating a STOP signal to terminate the communication, as shown in Figure 24-10. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

Table 30-8. CSR Field Descriptions (continued)

Bit	Name	Description
9–8	BTB	Branch target bytes. Defines the number of bytes of branch target address DDATA displays. 00 0 bytes 01 Lower 2 bytes of the target address 10 Lower 3 bytes of the target address 11 Entire 4-byte target address See Section 30.3.1, “Begin Execution of Taken Branch (PST = 0x5).”
7	—	Reserved, should be cleared.
6	NPL	Non-pipelined mode. Determines whether the core operates in pipelined or mode or not. 0 Pipelined mode 1 Nonpipelined mode. The processor effectively executes one instruction at a time with no overlap. This adds at least 5 cycles to the execution time of each instruction. Given an average execution latency of 1.6 cycles/instruction, throughput in non-pipeline mode would be 6.6 cycles/instruction, approximately 25% or less of pipelined performance. Regardless of the NPL state, a triggered PC breakpoint is always reported before the triggering instruction executes. In normal pipeline operation, the occurrence of an address and/or data breakpoint trigger is imprecise. In non-pipeline mode, triggers are always reported before the next instruction begins execution and trigger reporting can be considered precise.
5	IPI	Ignore pending interrupts. 1 Core ignores any pending interrupt requests signalled while in single-instruction-step mode. 0 Core services any pending interrupt requests that were signalled while in single-step mode.
4	SSM	Single-step mode. Setting SSM puts the processor in single-step mode. 0 Normal mode. 1 Single-step mode. The processor halts after execution of each instruction. While halted, any BDM command can be executed. On receipt of the GO command, the processor executes the next instruction and halts again. This process continues until SSM is cleared.
3–0	—	Reserved, should be cleared.

30.4.5 Data Breakpoint/Mask Registers (DBR, DBMR)

The DBR, shown in Figure 30-8, specifies data patterns used as part of the trigger into debug mode. DBR bits are masked by setting corresponding DBMR bits, as defined in TDR.

	31 … 0
Field	Data (DBR); Mask (DBMR)
Reset	Uninitialized
R/W	DBR is accessible in supervisor mode as debug control register 0x0E, using the WDEBUG instruction and through the BDM port using the RDMREG and WDMREG commands. DBMR is accessible in supervisor mode as debug control register 0x0F,using the WDEBUG instruction and via the BDM port using the WDMREG command.
DRc[4–0]	0x0E (DBR), 0x0F (DBMR)

Figure 30-8. Data Breakpoint/Mask Registers (DBR/DBMR)